Integrated Planning for Telepresence with Time Delays

Mark D. Johnston & Kenneth J. Rabe

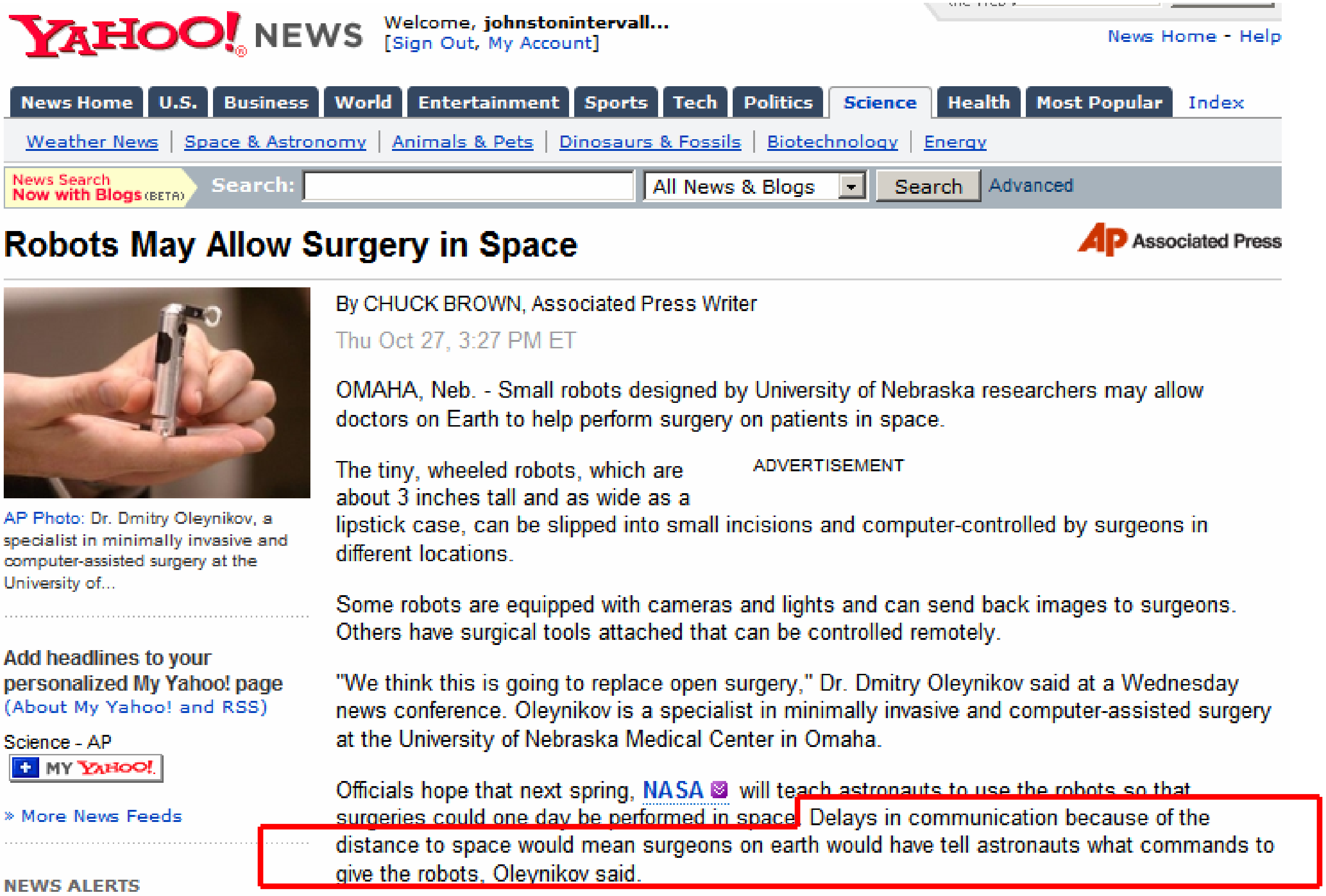
JPL/California Institute of Technology
4800 Oak Grove Drive
Pasadena CA 91109

The Problem

- Teleoperation of humanoid robotics with medium-range time delays
  - many DOF
  - 1.5-10s round-trip delay, e.g. for lunar operations
  - delay can introduce instability
- Various approaches have been studied and used:
  - “bump and wait”
  - predictive display
  - stabilized bilateral control
  - supervisory control

  but the problem is far from solved…

Future Robot Surgery?


YAHOO! NEWS
Welcome, johnstonintervall...
[Sign Out, My Account]
News Home - Help
News Home | U.S. | Business | World | Entertainment | Sports | Tech | Politics | Science | Health | Most Popular | Index
Weather News | Space & Astronomy | Animals & Pets | Dinosaurs & Fossils | Biotechnology | Energy
News Search Now with Blogs (BETA)
Search:
All News & Blogs
Search
Advanced
AP Associated Press

Robots May Allow Surgery in Space

By CHUCK BROWN, Associated Press Writer

Thu Oct 27, 3:27 PM ET

AP Photo: Dr. Dmitry Oleynikov, a specialist in minimally invasive and computer-assisted surgery at the University of...

OMAHA, Neb. - Small robots designed by University of Nebraska researchers may allow doctors on Earth to help perform surgery on patients in space.

ADVERTISEMENT

The tiny, wheeled robots, which are about 3 inches tall and as wide as a lipstick case, can be slipped into small incisions and computer-controlled by surgeons in different locations.

Some robots are equipped with cameras and lights and can send back images to surgeons. Others have surgical tools attached that can be controlled remotely.

"We think this is going to replace open surgery," Dr. Dmitry Oleynikov said at a Wednesday news conference. Oleynikov is a specialist in minimally invasive and computer-assisted surgery at the University of Nebraska Medical Center in Omaha.

Officials hope that next spring, NASA will teach astronauts to use the robots so that surgeries could one day be performed in space. Delays in communication because of the distance to space would mean surgeons on earth would have tell astronauts what commands to give the robots, Oleynikov said.

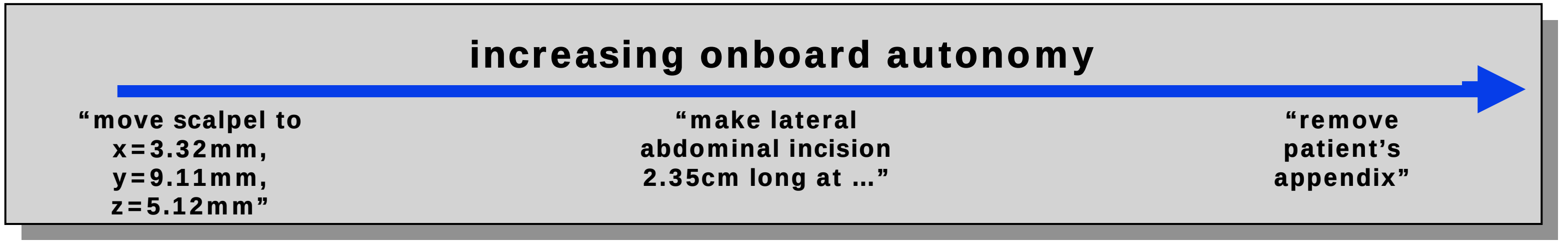
Add headlines to your personalized My Yahoo! page
(About My Yahoo! and RSS)

Science - AP

+ MY YAHOO!

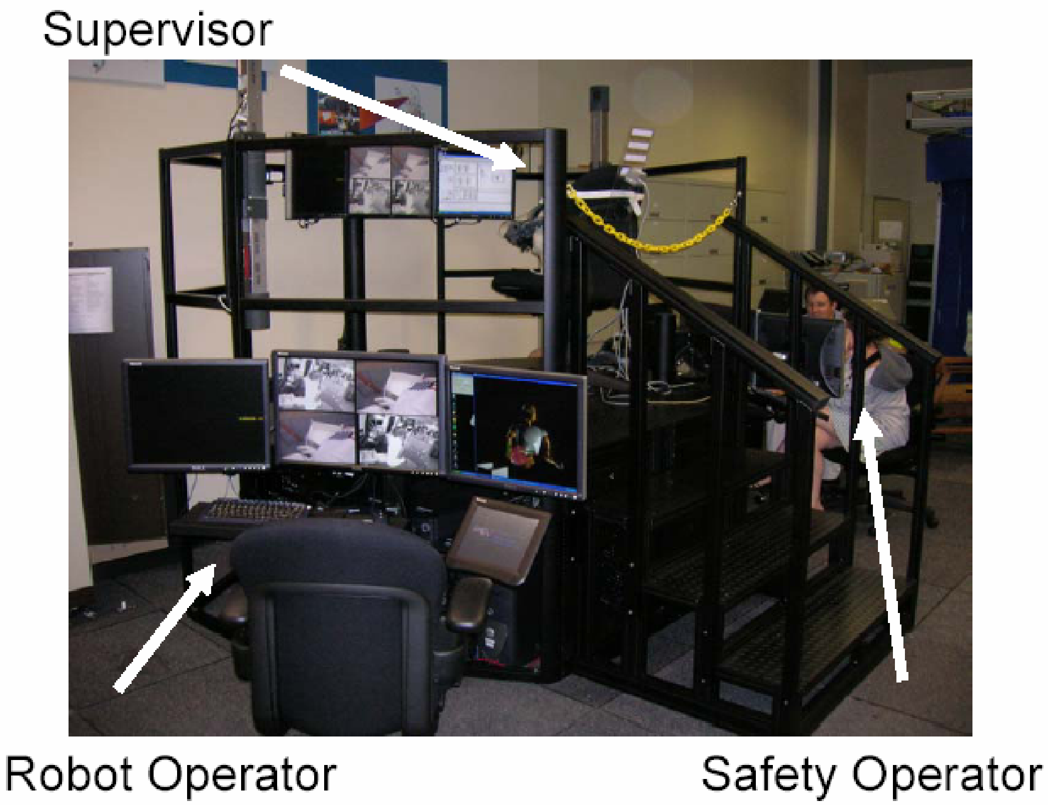
» More News Feeds

NEWS ALERTS


increasing onboard autonomy
"move scalpel to x=3.32mm, y=9.11mm, z=5.12mm"
"make lateral abdominal incision 2.35cm long at …"
"remove patient's appendix"

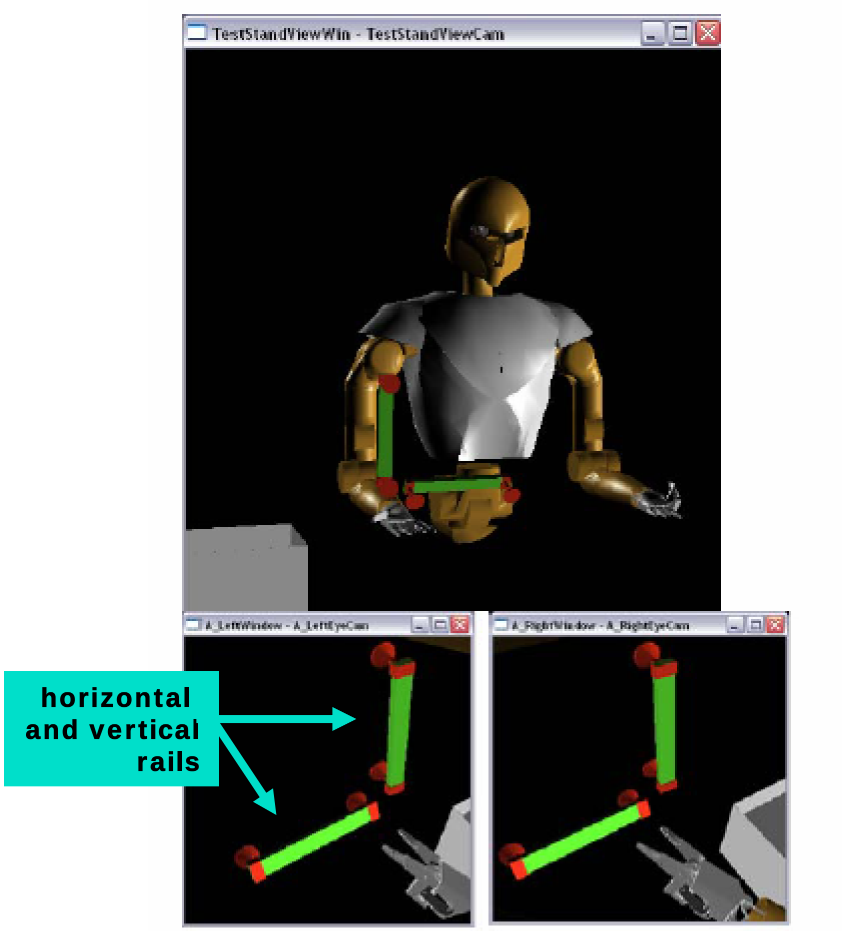
Approach — Overview

- Supervisor guides robot from “smart cockpit” in an immersive “virtual world” (JSC)
- Task-level assistant (JPL) guides the supervisor on task sequences
- Supervisor’s motions are monitored by Operator Intent Prediction (ARC) software that interprets motions as symbolic commands
- The remote robot can autonomously execute these symbolic commands



Cockpit

Robonaut


TestStandViewWin - TestStandViewCam
A_LeftWindow - A_LeftEyeCam
A_RightWindow - A_RightEyeCam

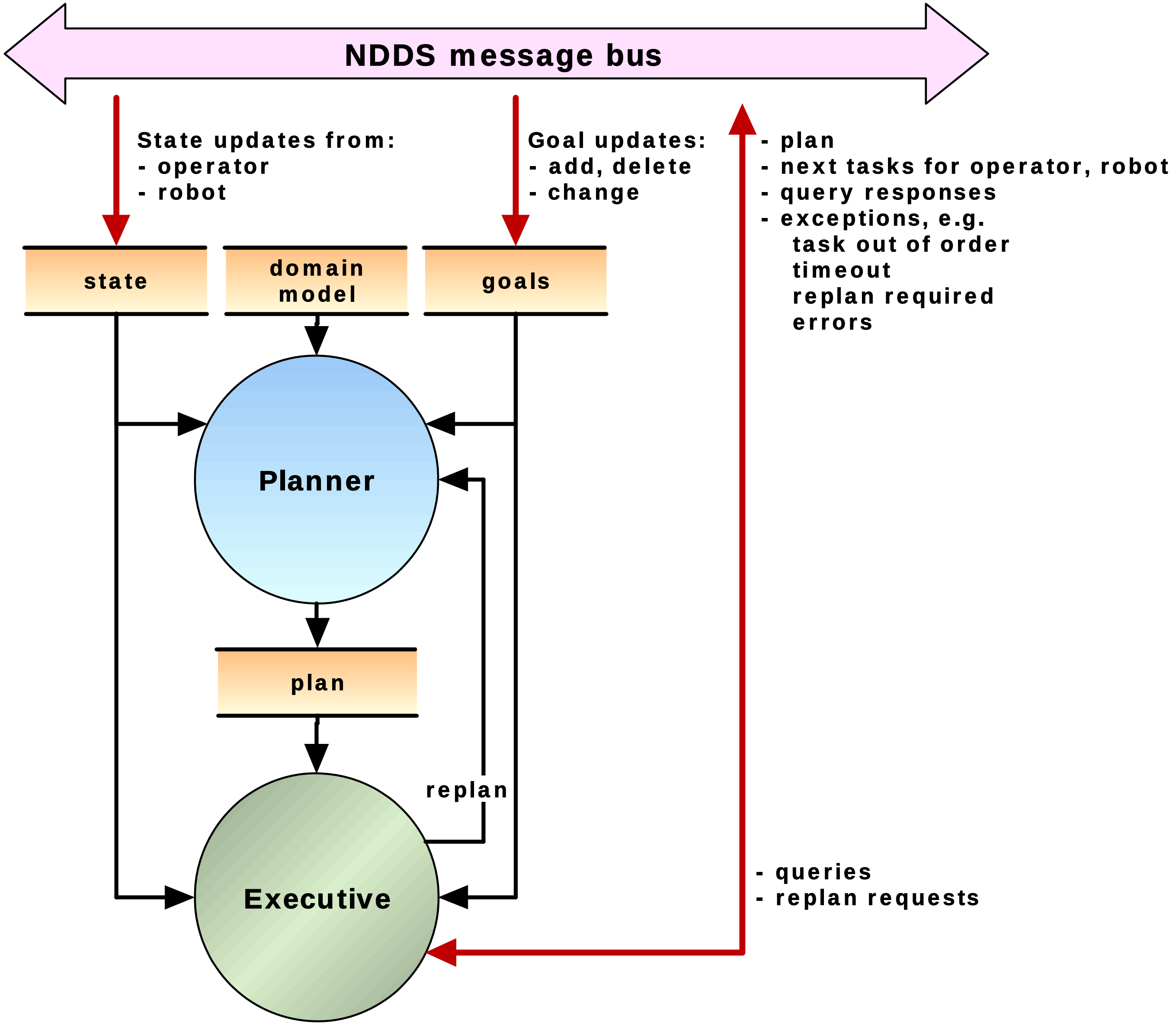
horizontal and vertical rails

Normal Planning and Execution

- Planning software generates task-level plan to achieve high level goals
- Human supervisor executes plan steps
- Robot follows, after time delay
- Planner/Executive software monitors:
  - supervisor and robot actions
  - state changes
  - goal changes

and *responds appropriately to any change*

Planner Context



State updates from:
- operator
- robot

Goal updates:
- add, delete
- change

- plan
- next tasks for operator, robot
- query responses
- exceptions, e.g.
  task out of order
  timeout
  replan required
  errors

- queries
- replan requests

Planner Context (cont.)

- The supervisor executing the plan is an experienced human operator who may freely deviate from plan if desired
- There are dual feedback paths:
  - feedback from supervisor is nearly immediate and can be used to decide on next execution steps to recommend to supervisor
  - but:
    - a task is not complete until the robot executes it
    - feedback from robot is time-delayed (up to ~10s)
    - waiting to check preconditions reduces execution to “bump and wait”

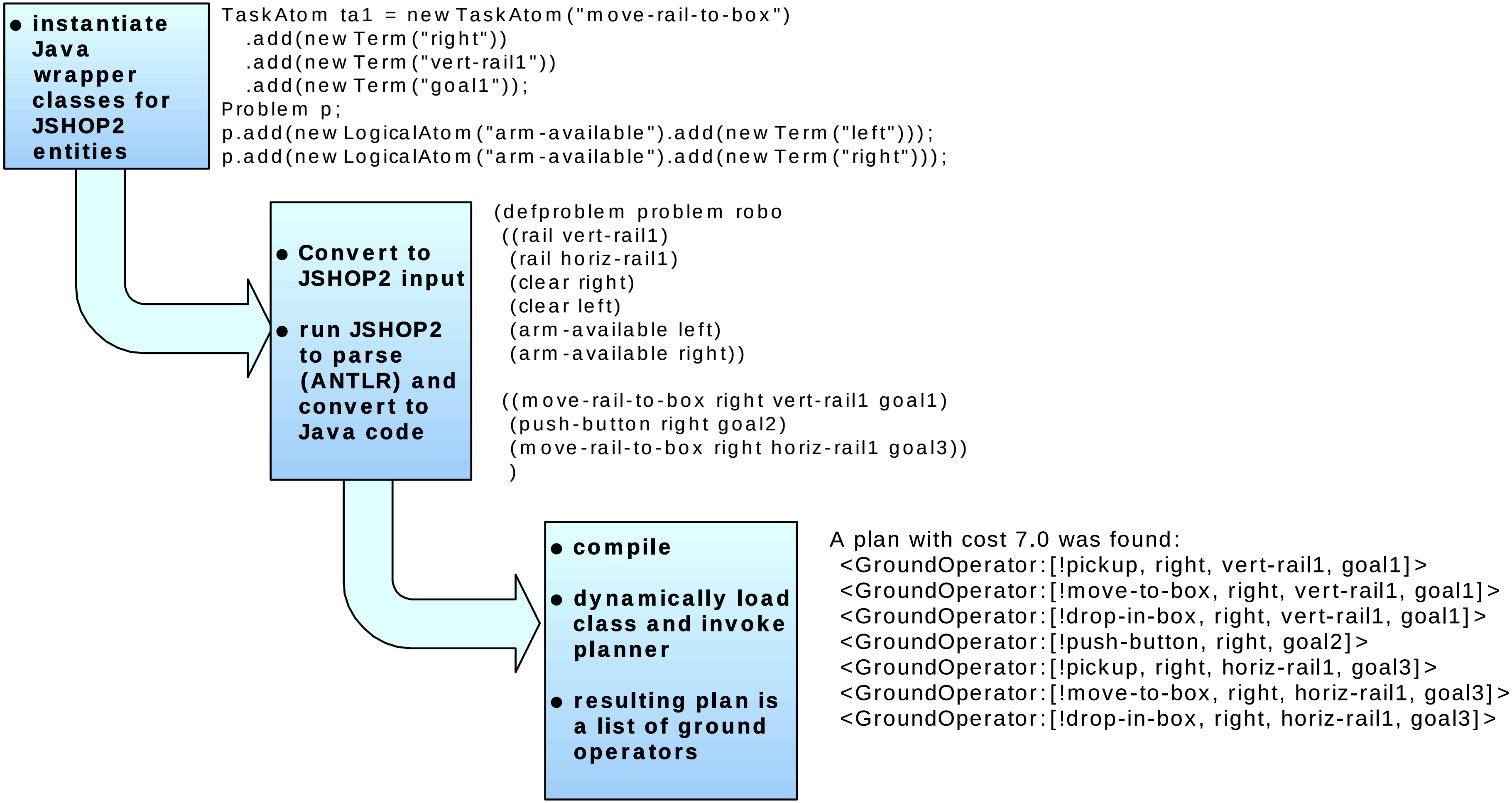
⇒ a significant challenge for the plan executive

Implementation

- Ensemble framework (Java)
- JNI NDDS interface layer (Win32)
- JSHOP2 planner
  - HTN planner, Java implementation
  - Nau *et al.*, http://www.cs.umd.edu/projects/shop
  - open source
- Piccolo ZUI toolkit (for testing GUI)
  - Bederson *et al.* http://www.cs.umd.edu/hcil/piccolo/index.shtml

Use of JSHOP2

- Phases of planning/replanning with JSHOP2 — from domain model + state + goals to a task sequence to achieve the goals:



- instantiate Java wrapper classes for JSHOP2 entities

```
TaskAtom ta1 = new TaskAtom("move-rail-to-box")
   .add(new Term("right"))
   .add(new Term("vert-rail1"))
   .add(new Term("goal1"));
Problem p;
p.add(new LogicalAtom("arm-available").add(new Term("left")));
p.add(new LogicalAtom("arm-available").add(new Term("right")));
```

- Convert to JSHOP2 input

- run JSHOP2 to parse (ANTLR) and convert to Java code

```
(defproblem problem robo
 ((rail vert-rail1)
  (rail horiz-rail1)
  (clear right)
  (clear left)
  (arm-available left)
  (arm-available right))

 ((move-rail-to-box right vert-rail1 goal1)
  (push-button right goal2)
  (move-rail-to-box right horiz-rail1 goal3))
  )
```

- compile

- dynamically load class and invoke planner

- resulting plan is a list of ground operators

```
A plan with cost 7.0 was found:
 <GroundOperator:[!pickup, right, vert-rail1, goal1]>
 <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>
 <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>
 <GroundOperator:[!push-button, right, goal2]>
 <GroundOperator:[!pickup, right, horiz-rail1, goal3]>
 <GroundOperator:[!move-to-box, right, horiz-rail1, goal3]>
 <GroundOperator:[!drop-in-box, right, horiz-rail1, goal3]>
```

Planning & Execution

Scenarios

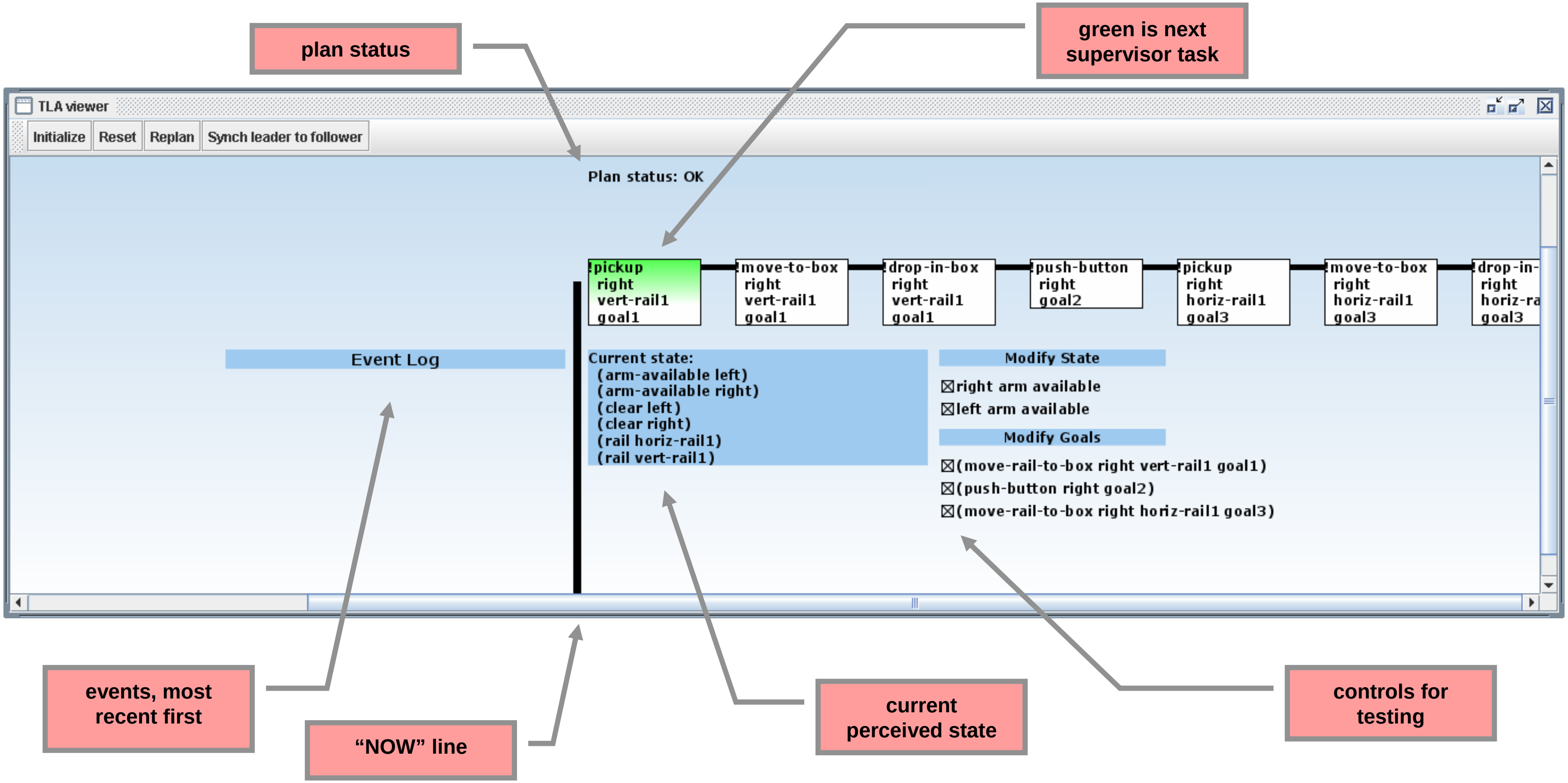
- normal step-by-step operation
- timed out robot response
- supervisor working ahead of robot (pipelining)
- deviations from plan
  - robot capability change
  - supervisor deviates unintentionally
  - supervisor deviates intentionally

Considerations

- handle replans gracefully
  - fall back to confirmed state
  - recognize accomplished goals
  - maintain plan continuity
- handle errors gracefully

Monitoring and Testing GUI

- Purpose:
  - visualize plan state
  - test planner under anticipated conditions



Plan status: OK

pickup right vert-rail1 goal1	move-to-box right vert-rail1 goal1	drop-in-box right vert-rail1 goal1	push-button right goal2	pickup right horiz-rail1 goal3	move-to-box right horiz-rail1 goal3	drop-in- right horiz-ra goal3

Current state:
(arm-available left)
(arm-available right)
(clear left)
(clear right)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

☒right arm available
☒left arm available

Modify Goals

☒(move-rail-to-box right vert-rail1 goal1)
☒(push-button right goal2)
☒(move-rail-to-box right horiz-rail1 goal3)

plan status

green is next supervisor task

events, most recent first

“NOW” line

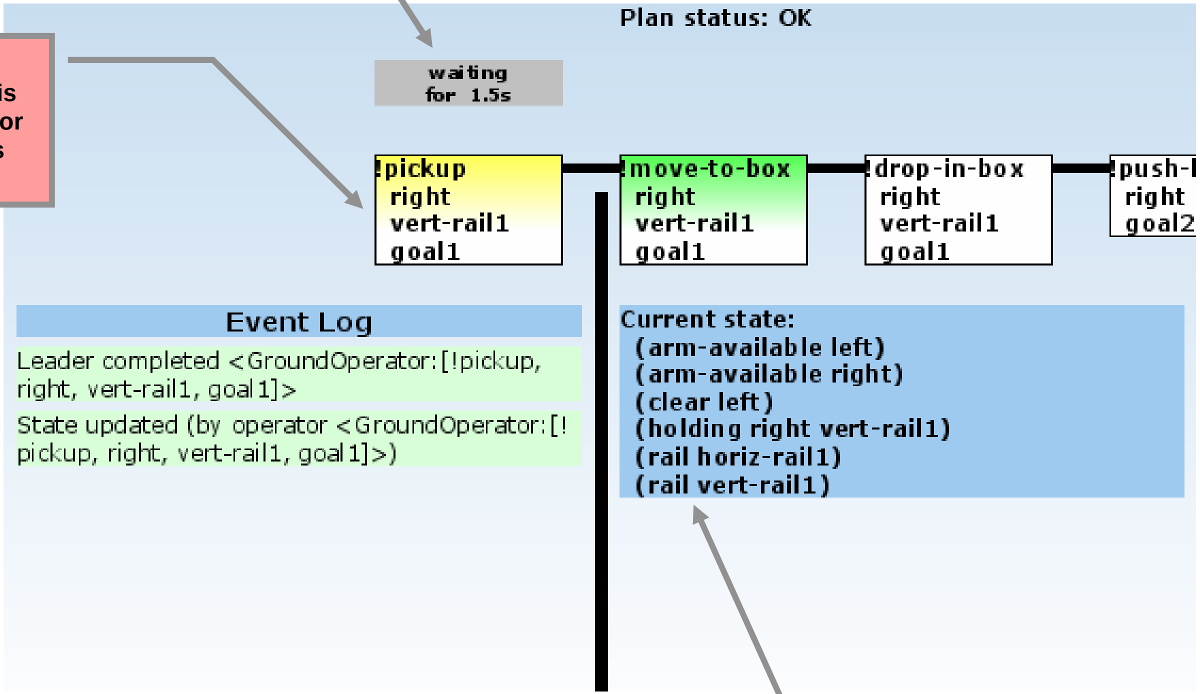
current perceived state

controls for testing

Normal sequence: first the supervisor acts…

time waiting

supervisor completed this task, waiting for robot across time delay



Plan status: OK

waiting
for 1.5s

!pickup
right
vert-rail1
goal1

!move-to-box
right
vert-rail1
goal1

!drop-in-box
right
vert-rail1
goal1

!push-l
right
goal2

Event Log

Leader completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>

State updated (by operator <GroundOperator:[!pickup, right, vert-rail1, goal1]>)

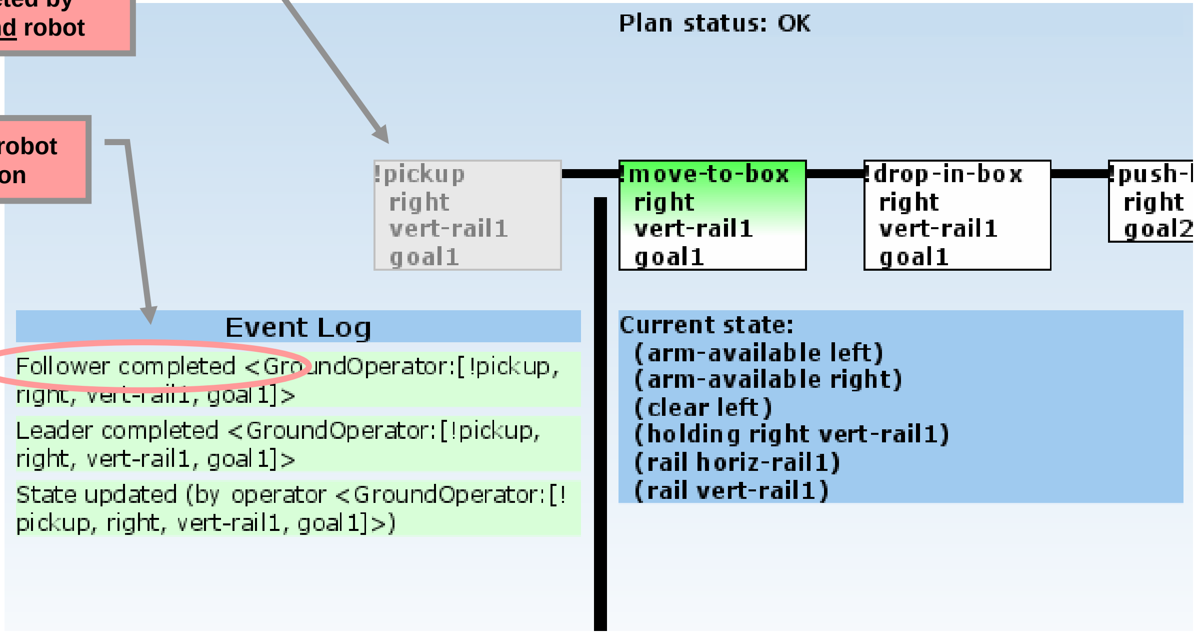
Current state:
(arm-available left)
(arm-available right)
(clear left)
(holding right vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

state updates to reflect supervisor action

…then the robot

grayed-out tasks have been completed by supervisor <u>and</u> robot

confirmation from robot of task completion



Plan status: OK

```
!pickup        !move-to-box    !drop-in-box    !push-l
 right          right           right           right
 vert-rail1     vert-rail1      vert-rail1      goal2
 goal1          goal1           goal1
```

Event Log

Follower completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>

Leader completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>

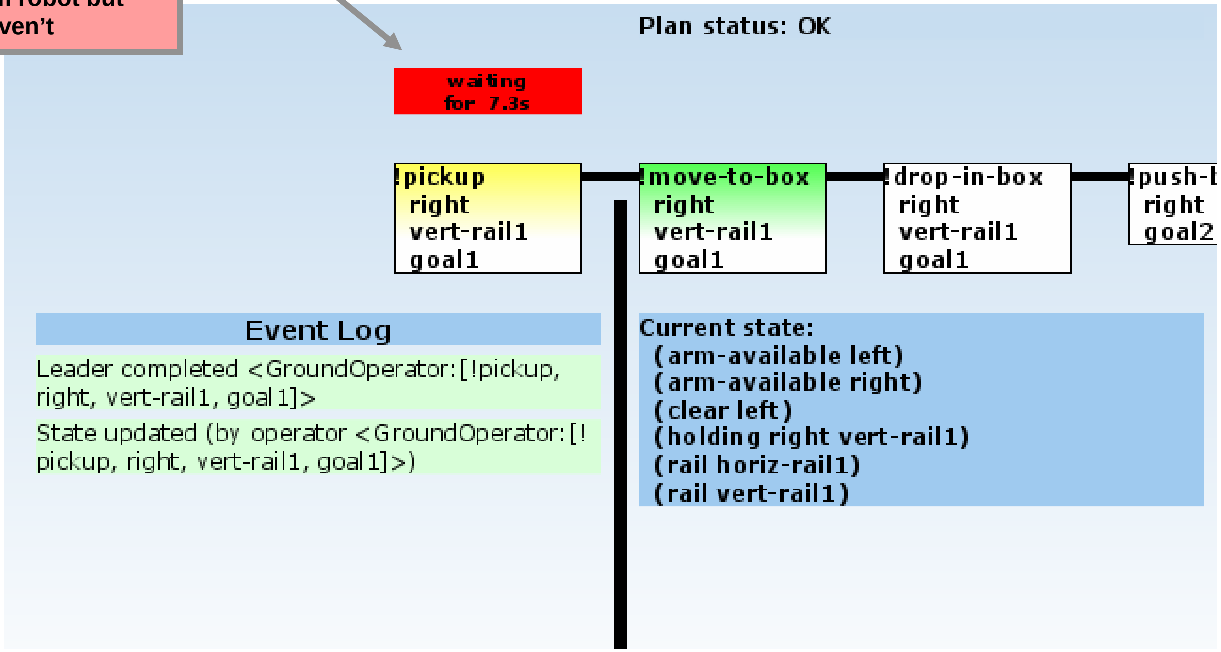
State updated (by operator <GroundOperator:[!pickup, right, vert-rail1, goal1]>)

```
Current state:
 (arm-available left)
 (arm-available right)
 (clear left)
 (holding right vert-rail1)
 (rail horiz-rail1)
 (rail vert-rail1)
```

Robot might be late

- a reason for an alert (here triggered after 6s)

timeout: expected to hear from robot but haven’t


Plan status: OK
waiting for 7.3s
!pickup right vert-rail1 goal1
!move-to-box right vert-rail1 goal1
!drop-in-box right vert-rail1 goal1
!push-b right goal2

Event Log

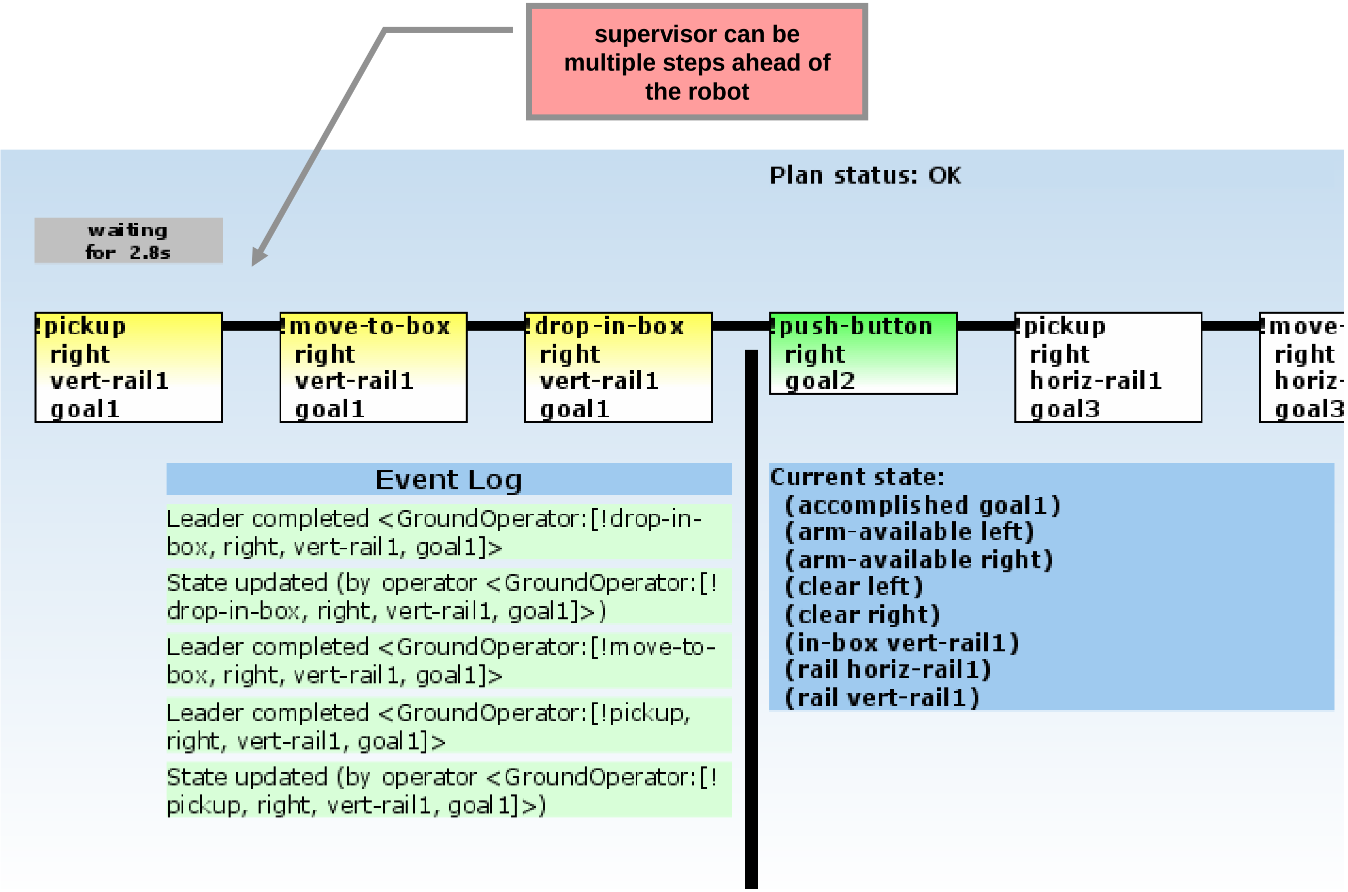
Leader completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>

State updated (by operator <GroundOperator:[!pickup, right, vert-rail1, goal1]>)

```
Current state:
  (arm-available left)
  (arm-available right)
  (clear left)
  (holding right vert-rail1)
  (rail horiz-rail1)
  (rail vert-rail1)
```

Supervisor can work ahead

- this “pipelining” is the basis for improved efficiency



supervisor can be multiple steps ahead of the robot

Plan status: OK

waiting for 2.8s

!pickup right vert-rail1 goal1

!move-to-box right vert-rail1 goal1

!drop-in-box right vert-rail1 goal1

!push-button right goal2

!pickup right horiz-rail1 goal3

!move- right horiz- goal3

Event Log

Leader completed <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>

State updated (by operator <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>)

Leader completed <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>

Leader completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>

State updated (by operator <GroundOperator:[!pickup, right, vert-rail1, goal1]>)

Current state:
(accomplished goal1)
(arm-available left)
(arm-available right)
(clear left)
(clear right)
(in-box vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

Deviations from Plan

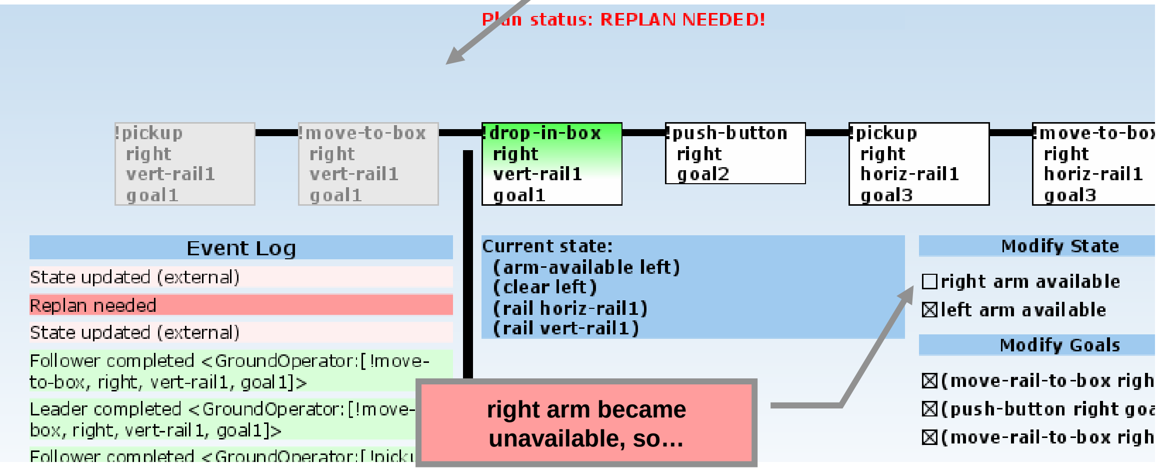
- Time out – waiting longer than expected
  - alert user, but take no other action
- Robot capability change
  - significant change that impacts plan
- Supervisor deviates from plan
  - unintentionally
  - intentionally
- Notes:
  - for this project we adopted a design principle of no automatic replans: replans must be requested by the supervisor

Robot State Change Example

Preferred arm becomes unavailable:

- state change, i.e. `(arm-available right)` retracted, triggers need to replan
- replan finds alternative plan using available resources: left arm instead of right

robot had completed two tasks when…



Plan status: REPLAN NEEDED!

!pickup right vert-rail1 goal1

!move-to-box right vert-rail1 goal1

!drop-in-box right vert-rail1 goal1

!push-button right goal2

!pickup right horiz-rail1 goal3

!move-to-box right horiz-rail1 goal3

Event Log

State updated (external)

Replan needed

State updated (external)

Follower completed <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>

Leader completed <GroundOperator:[!move-box, right, vert-rail1, goal1]>

Follower completed <GroundOperator:[!picku

Current state:
(arm-available left)
(clear left)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

☐right arm available

☒left arm available

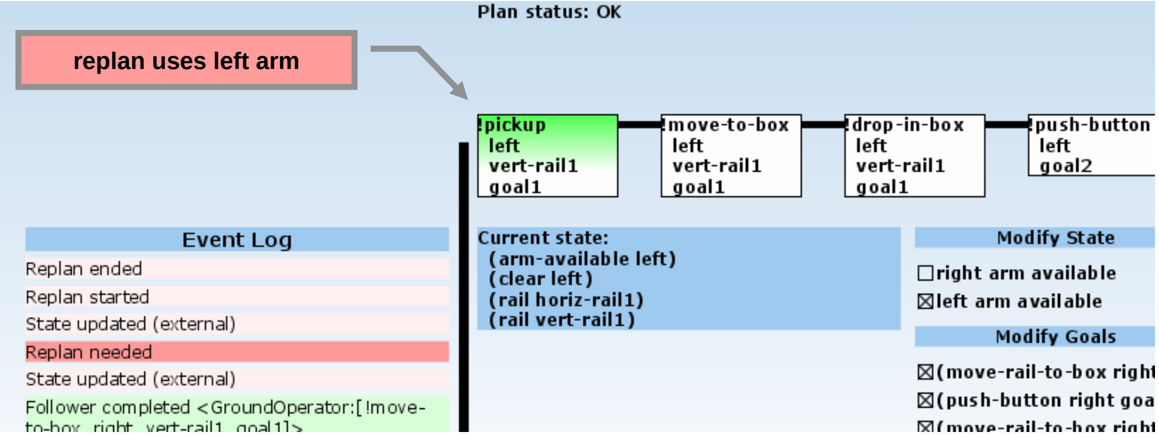
Modify Goals

☒(move-rail-to-box righ

☒(push-button right goa

☒(move-rail-to-box righ

right arm became unavailable, so…



Plan status: OK

replan uses left arm

!pickup left vert-rail1 goal1

!move-to-box left vert-rail1 goal1

!drop-in-box left vert-rail1 goal1

!push-button left goal2

Event Log

Replan ended

Replan started

State updated (external)

Replan needed

State updated (external)

Follower completed <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>

Current state:
(arm-available left)
(clear left)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

☐right arm available

☒left arm available

Modify Goals

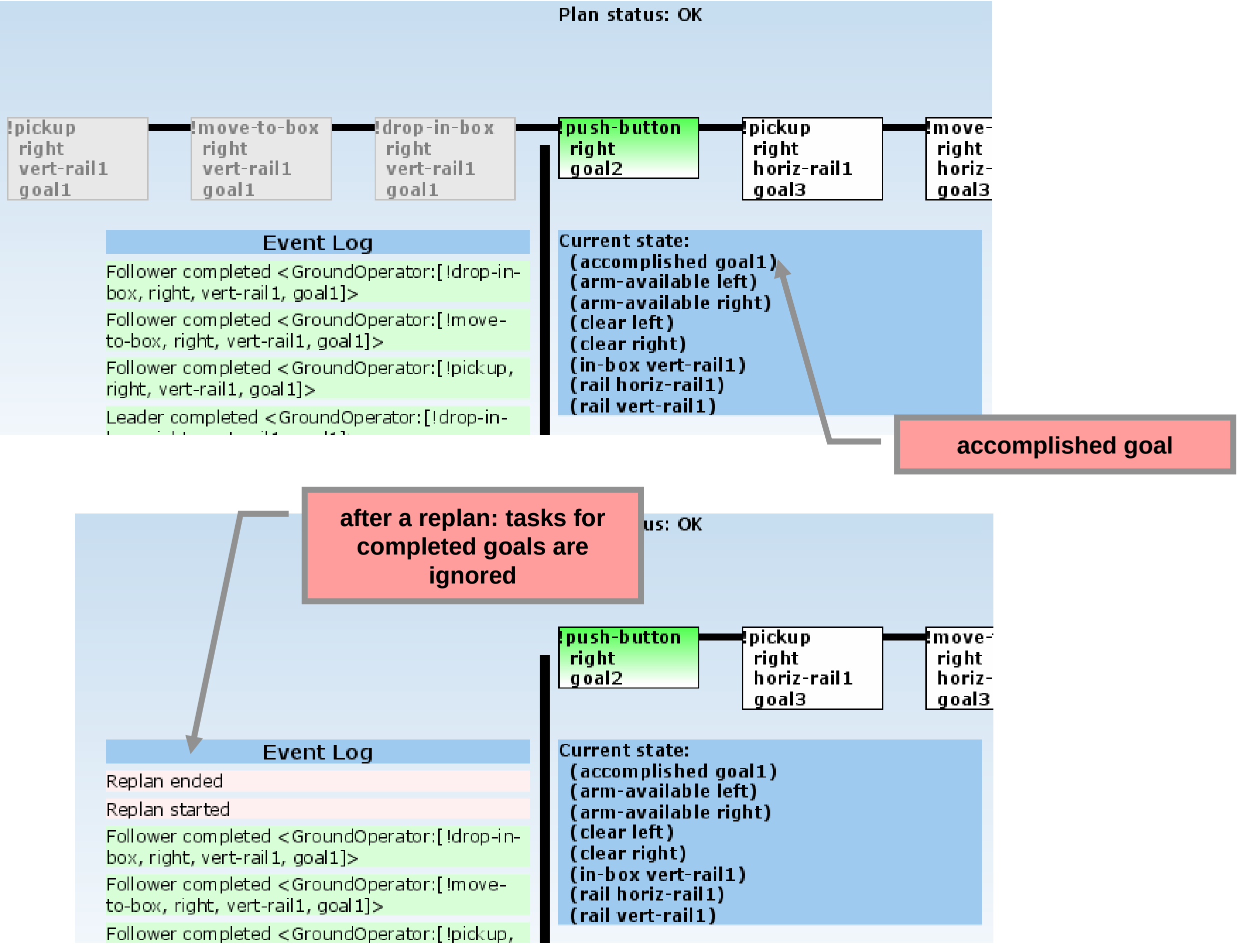
☒(move-rail-to-box right

☒(push-button right goa

☒(move-rail-to-box right

Accomplished goals skipped in replan

- replanning takes account of current state
- accomplished goals are ignored in a replan



Plan status: OK

!pickup right vert-rail1 goal1
!move-to-box right vert-rail1 goal1
!drop-in-box right vert-rail1 goal1
!push-button right goal2
!pickup right horiz-rail1 goal3
!move- right horiz- goal3

Event Log

```
Follower completed <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>
Follower completed <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>
Follower completed <GroundOperator:[!pickup, right, vert-rail1, goal1]>
Leader completed <GroundOperator:[!drop-in-
```

```
Current state:
 (accomplished goal1)
 (arm-available left)
 (arm-available right)
 (clear left)
 (clear right)
 (in-box vert-rail1)
 (rail horiz-rail1)
 (rail vert-rail1)
```

accomplished goal

after a replan: tasks for completed goals are ignored

us: OK

!push-button right goal2
!pickup right horiz-rail1 goal3
!move- right horiz- goal3

Event Log

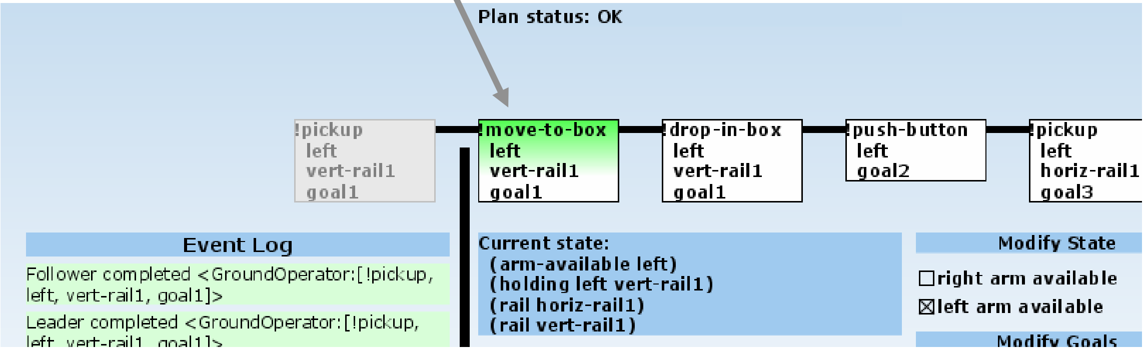
```
Replan ended
Replan started
Follower completed <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>
Follower completed <GroundOperator:[!move-to-box, right, vert-rail1, goal1]>
Follower completed <GroundOperator:[!pickup,
```

```
Current state:
 (accomplished goal1)
 (arm-available left)
 (arm-available right)
 (clear left)
 (clear right)
 (in-box vert-rail1)
 (rail horiz-rail1)
 (rail vert-rail1)
```

Planning continuity…

- If right arm returns while left arm is busy, the current left arm tasks are planned to completion:

if right arm returns while left arm is holding a rail


Plan status: OK

Event Log

Follower completed <GroundOperator:[!pickup, left, vert-rail1, goal1]>

Leader completed <GroundOperator:[!pickup, left, vert-rail1, goal1]>

Current state:
(arm-available left)
(holding left vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

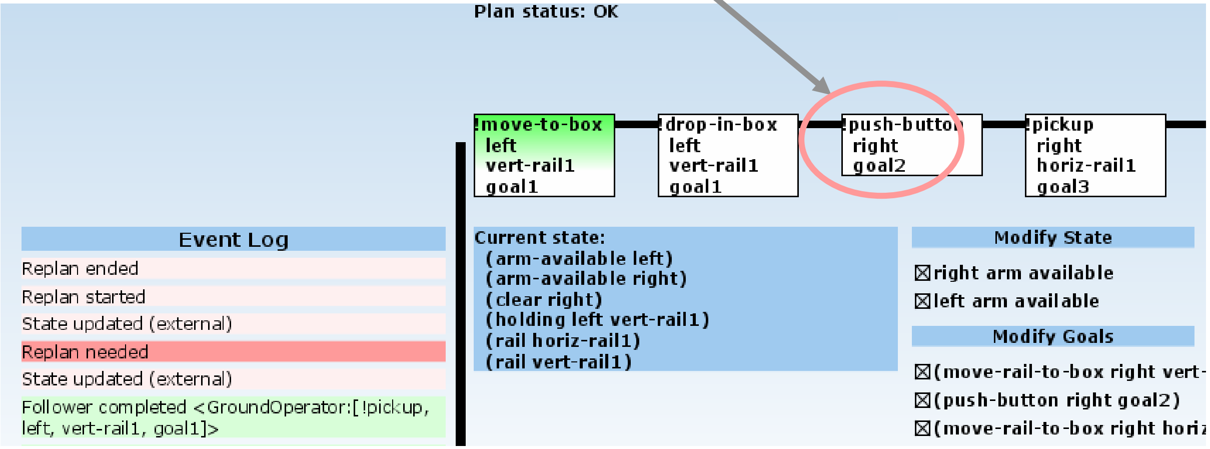
Modify State

□right arm available

☒left arm available

Modify Goals

replan starts using right only after left is finished


Plan status: OK

Event Log

Replan ended

Replan started

State updated (external)

Replan needed

State updated (external)

Follower completed <GroundOperator:[!pickup, left, vert-rail1, goal1]>

Current state:
(arm-available left)
(arm-available right)
(clear right)
(holding left vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

☒right arm available

☒left arm available

Modify Goals

☒(move-rail-to-box right vert-

☒(push-button right goal2)

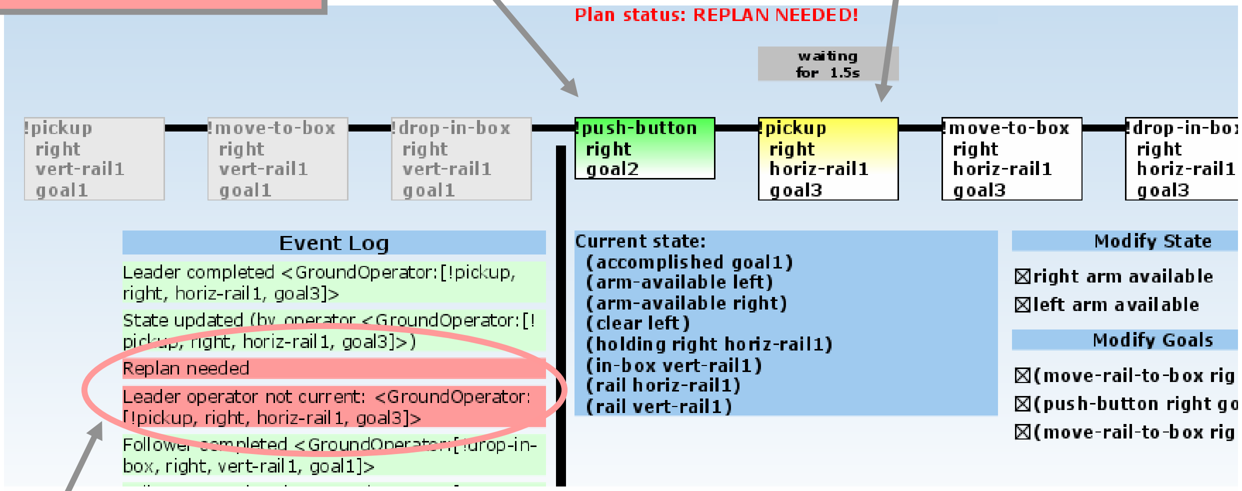
☒(move-rail-to-box right horiz

Supervisor Deviation From Plan

- Supervisor intention is expressed by the current goals, so any deviation is assumed to be *unintentional* unless the goals change

plan called for supervisor to push button…

but supervisor picked up rail instead


Plan status: REPLAN NEEDED!
waiting for 1.5s
!pickup right vert-rail1 goal1
!move-to-box right vert-rail1 goal1
!drop-in-box right vert-rail1 goal1
!push-button right goal2
!pickup right horiz-rail1 goal3
!move-to-box right horiz-rail1 goal3
!drop-in-box right horiz-rail1 goal3

Event Log

Leader completed <GroundOperator:[!pickup, right, horiz-rail1, goal3]>

State updated (by operator <GroundOperator:[!pickup, right, horiz-rail1, goal3]>)

Replan needed

Leader operator not current: <GroundOperator:[!pickup, right, horiz-rail1, goal3]>

Follower completed <GroundOperator:[!drop-in-box, right, vert-rail1, goal1]>

Current state:
(accomplished goal1)
(arm-available left)
(arm-available right)
(clear left)
(holding right horiz-rail1)
(in-box vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

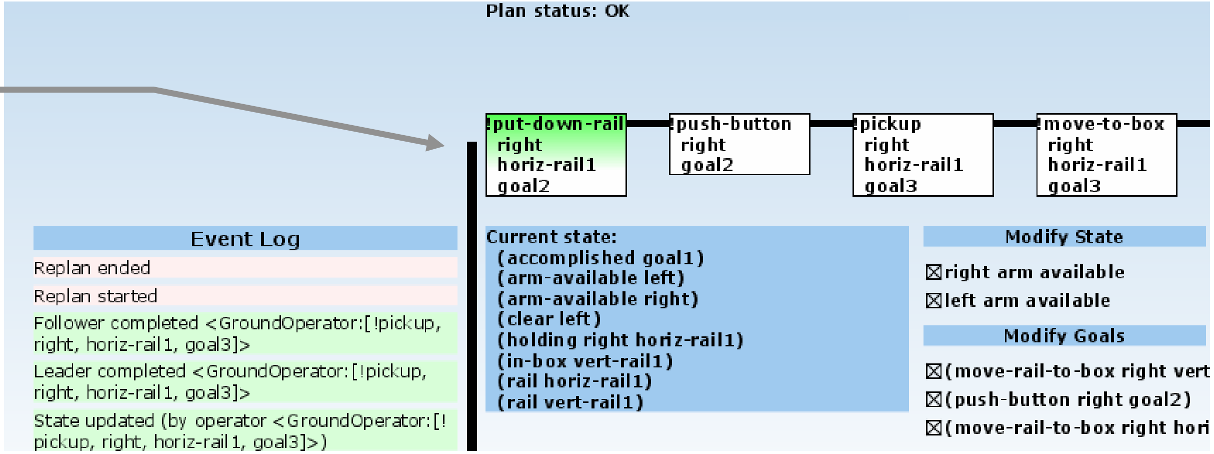
☒right arm available
☒left arm available

Modify Goals

☒(move-rail-to-box rig
☒(push-button right go
☒(move-rail-to-box rig

detected "out of order" operation

a replan at this point generates a plan to take steps to resume the previous plan


Plan status: OK
!put-down-rail right horiz-rail1 goal2
!push-button right goal2
!pickup right horiz-rail1 goal3
!move-to-box right horiz-rail1 goal3

Event Log

Replan ended

Replan started

Follower completed <GroundOperator:[!pickup, right, horiz-rail1, goal3]>

Leader completed <GroundOperator:[!pickup, right, horiz-rail1, goal3]>

State updated (by operator <GroundOperator:[!pickup, right, horiz-rail1, goal3]>)

Current state:
(accomplished goal1)
(arm-available left)
(arm-available right)
(clear left)
(holding right horiz-rail1)
(in-box vert-rail1)
(rail horiz-rail1)
(rail vert-rail1)

Modify State

☒right arm available
☒left arm available

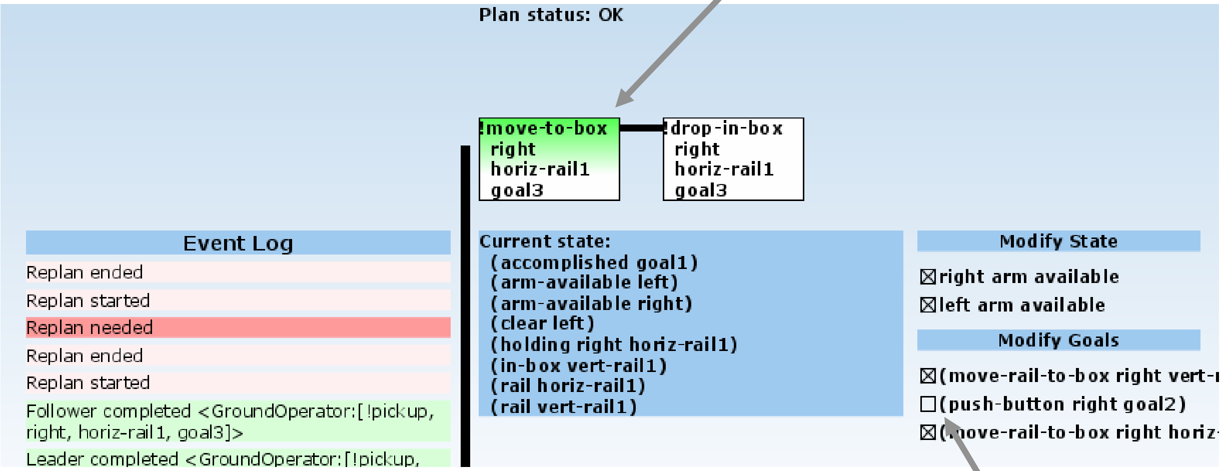
Modify Goals

☒(move-rail-to-box right vert
☒(push-button right goal2)
☒(move-rail-to-box right hori

Intentional Deviation

- If supervisor *intended* to skip the push button step, then the high-level goals need to be changed to reflect this
- Once the obsolete goal is removed, the replan corresponds to the supervisor’s intention

replanning after a goal change expresses the supervisor’s intention



Plan status: OK

!move-to-box
right
horiz-rail1
goal3

!drop-in-box
right
horiz-rail1
goal3

Event Log

Replan ended
Replan started
Replan needed
Replan ended
Replan started
Follower completed <GroundOperator:[!pickup, right, horiz-rail1, goal3]>
Leader completed <GroundOperator:[!pickup,

```
Current state:
 (accomplished goal1)
 (arm-available left)
 (arm-available right)
 (clear left)
 (holding right horiz-rail1)
 (in-box vert-rail1)
 (rail horiz-rail1)
 (rail vert-rail1)
```

Modify State

☒right arm available
☒left arm available

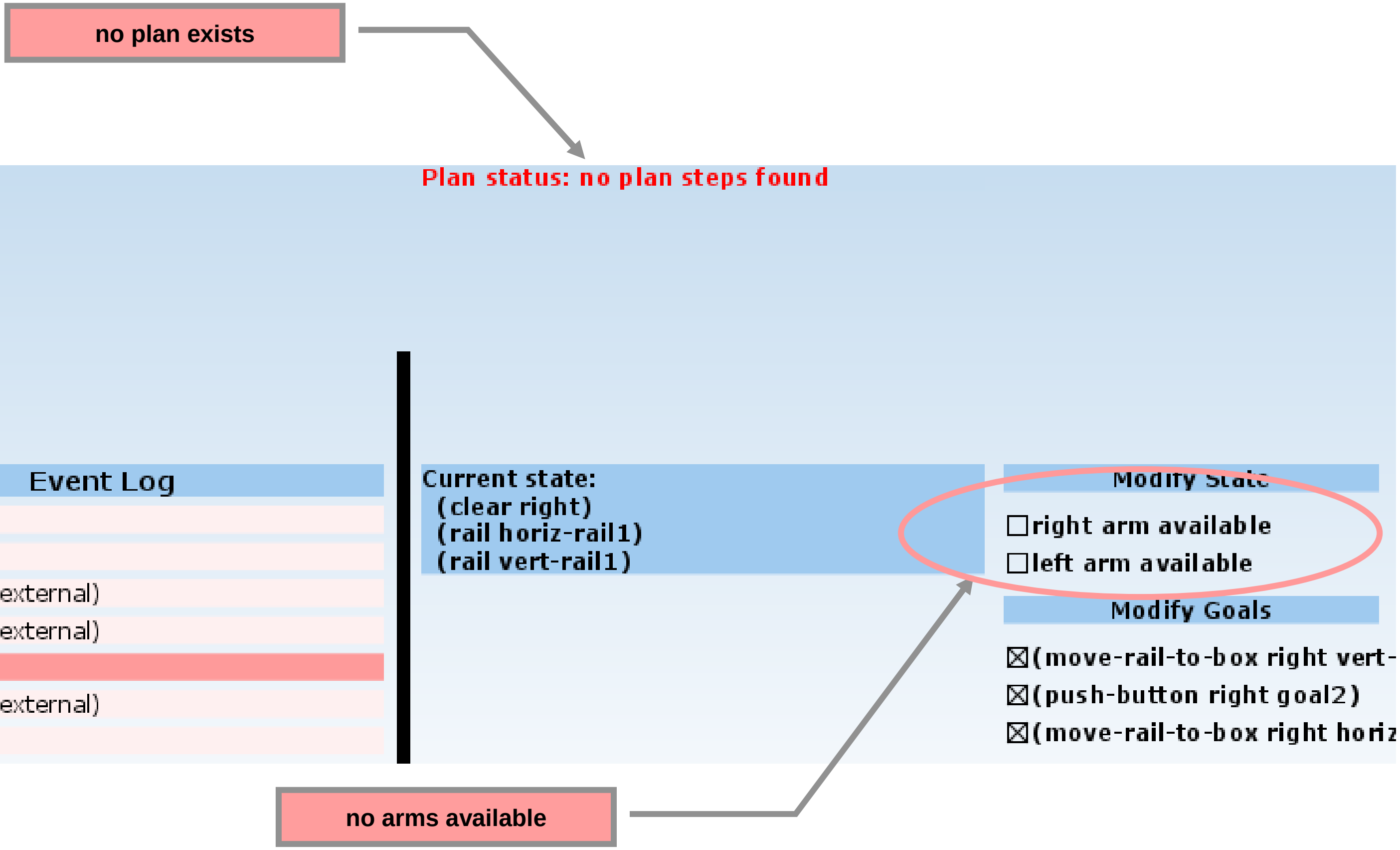
Modify Goals

☒(move-rail-to-box right vert-r
☐(push-button right goal2)
☒(move-rail-to-box right horiz-

removed goal

Infeasible states

- State changes can obviously make finding a plan impossible



no plan exists

Plan status: no plan steps found

Event Log

Replan ended
Replan started
State updated (external)
State updated (external)
Replan needed
State updated (external)
Replan ended

Current state:
 (clear right)
 (rail horiz-rail1)
 (rail vert-rail1)

Modify State

☐right arm available
☐left arm available

Modify Goals

☒(move-rail-to-box right vert-
☒(push-button right goal2)
☒(move-rail-to-box right horiz

no arms available

Concluding Comments

- For the scope of this project, JSHOP2 has been sufficient for all needs
  - fast simple modeling
  - HTN “plan in execution order” is natural for human supervisor
  - but, clumsy to map into Lisp syntax for processing
  - Temporal constraints are not an issue here, but could be in a more complex domain
    - will require either additional modeling effort, or migration to a planner that supports temporal reasoning from the outset
- The leader/follower model for planning and execution has proven a good fit for the time-delayed telepresence problem
  - anticipate applying the system to demonstration hardware later in 2006